SAND91 - 0829

Distribution
Category UC-126

Unlimited Release
Printed July 1991

A USER'S GUIDE TO A SIMPLIFIED CREEP MODEL FOR CAVERNS IN THE STRATEGIC PETROLEUM RESERVE

Grant S. Heffelfinger
Underground Storage Technology Division 6257
Sandia National Laboratories
Albuquerque, New Mexico 87185

ABSTRACT

A FORTRAN computer program has been developed to model creep closure in Strategic Petroleum Reserve (SPR) caverns. These caverns, leached in Gulf Coast salt domes, contain a reserve of nearly 750-million barrels of crude oil which is maintained by the U.S. Department of Energy. Historically, most workers modelling creep closure in SPR caverns and similar salt dome cavities have employed finite element methods. This simplified creep model differs from these methods in that it is not only able to predict volume loss due to creep closure, but also the evolution with time of wellhead pressure and oil/brine interface depth, as well as the relationships between rate of fluid removal or leakage and operational parameters. In addition, this simplified model is much less computationally intensive than the traditional methods. This report, a companion to an earlier report which documented the model's mathematics, computer algorithm, and initial results, documents the information necessary to run the program and interpret the results and includes the program's source code.

INTRODUCTION

To develop a creep closure model which could not only predict volume loss but also changes in wellhead pressure, oil/brine interface depth, and fluid removal rates with time, a simplified approach was taken yielding a model that is much less computationally intensive than standard techniques (such as finite element analysis). The mathematics and algorithm of this model have been documented1 and it is the purpose of this document to guide users of the simplified model, both in running the program and in interpreting the results. In addition the FORTRAN source code of the model is included in Appendix A.

The program was written with the goal of producing a user friendly computer model that not only modelled the behavior of actual SPR caverns but also produced data in a format which closely resembled that of actual caverns. In addition, this program was intended to be run on microcomputers. Most of the SPR cavern modelling with this program to data has been done on a Sun 386i workstation, requiring approximately 2 hours for a 30 year simulation with a time step of 0.05 years.

PROGRAM DESIGN

As discussed in the Introduction, a primary goal of the model design was to build an easy-to-use computer program that mimicked actual caverns, both in behavior as well as in output data format. Thus the program has been written to run interactively, with the user controlling cavern operation from the terminal. For longer calculations, the user's control inputs can be supplied by an input file. In addition, the program can automatically simulate normal cavern operation without user interaction by calculating brine removal rates which achieve the operational goal of keeping the wellhead pressure constant.

The program requires one input file containing the model parameters such as cavern depth, initial wellhead pressure, etc., as well as the program's control parameters, such as the time step to be used for the calculations. This input file is designated "sprcreep.1" by the FORTRAN open statement and is addressed by channel 1. A second input file, "sprcreep.7", addressed by channel 7, is required if a cavern with an initial geometry other than that of a right circular cylinder is used or if the calculations are to begin from previous program output. If the program is not run at constant wellhead pressure, the user must input when and how much brine and/or oil to remove. This can be done interactively or with a third input file, "sprcreep.9", addressed by channel 9. These input files are addressed individually in the discussion below.

The program generates three output files. The FORTRAN channel 2 is defined by an open statement to be "sprcreep.2" and contains the input parameters (from "sprcreep.1") followed by information generated at each time step. The channel 3 definition, "sprcreep.3" contains information,

including cavern radius as a function of depth, r(z), which is written to the file at evenly spaced increments in time, as specified by the user. Finally, **"sprcreep.8",** the FORTRAN channel 8 output file, contains an updated record of r(z). This record can be used as a channel 7 input to restart the program if it should terminate prematurely due to hardware errors. These output files are also discussed more completely below.

Input Files

"sprcreep.1"

An example of this file has been included in Fig. 1. The first two lines of this input file include the parameters: **"Pw",** "Rft", **"zt", "zb",** and **"zi0",** the initial **wellhead** pressure, the initial radius at the top of the cavern, the distance from the surface to the top of the cavern, the distance from the surface to the bottom of the cavern, and the initial distance from the surface to the oil/brine interface, respectively. The next three lines contain the program's control parameters. The program's time step in years, "deltat", should be taken to be 0.1 yrs or smaller for most 30 year calculations. This parameter can easily be tinkered with to determine the largest acceptable time step needed for a particular application. That is, by running the program with a very small **timestep** (for example, 0.005 - 0.01 years) followed by similar runs with larger timesteps, one can choose the largest value of "deltat" which gives acceptable deviations in results from those produced by running the program at the smallest timestep. For most uses, 0.05 years is sufficiently small. The parameter "tquit" is the time, in years, of the final calculation. This parameter is enforced regardless of the mode of operation. Thus the program will cease running at this time, even if it is running interactively. The parameter **"tdump"** is the time interval, in years, at which the program records the calculated r(z) profile as well as other parameters, in **"sprcreep.3"** and **"sprcreep.8".** The next two parameters, "LPtConst" and "Lterm" are FORTRAN logical parameters which are read with a formatted READ statement, thus these variables, whether **"T"** or **"F",** must be placed in the correct columns. If the program is running at constant **wellhead** pressure, "LPtConst" is set to **"T"** (true) and the program will remove an appropriate amount of brine every **"DtMb"** years. If the user desires a continuous oil leak (calculated every time **step), "bcleak"** defines the time, in years, at which the leak is to begin, **"ecleak"** is the time, in years, at which the leak is to end, and "Bcremove" is the leak rate in barrels of oil per time step. If "LPtConst" is **"F"** (false), the program will allow the **wellhead** pressure to vary according to inputs either from the user's terminal (if "Lterm" is "T") or from the file **"sprcreep.9"** (if "Lterm" is **"F").** This feature of the program is discussed more completely in the text that follows. The parameter **"spgr"** is simply the specific gravity of the oil. The next set of parameters in the **"sprcreep.1"** input file, **"Klitho",** "E", "mu", **"A_yr",** "Elast", and **"nu"** are defined in the two equations that follow, the creep and elastic response equations. The creep **equation**[1] is:

$$\Delta r_c = R\ A\ \Delta t\ \exp\left(\frac{-E}{R_g T}\right)\left\{\frac{\left|K_{litho}\ z - P(z)\right|}{\mu}\right\}^{5.5} \qquad (1)$$

Figure 1

"sprcreep.1" Input File for
a Run at Constant P_w

600	100.0		**Pw (psia)**	Rft (ft)	
2000	4000	3750	zt (ft)	zb (ft)	**zi0 (ft)**
0.05	1.0	0.50	deltat (yr)	tquit **(yr)**	tdump **(yr)**
0.10	0.35	0.85	**DtMb (yr)**	**bcleak** (yr)	ecleak (yr)
T	F		**LPtCONST**	Lterm	
150.0	0.876		Bcremove **(bbl/deltat)**		**spgr**
1.0	12581.78	**1.23188e6**	Klitho (psia/ft)	**E (K)**	mu (psia)
2.4309e+29			**A_yr (1/yr)**		
4.166e6	0.30		Elast (psia)	**nu**	
7.778e-3	299.44		**mT (K/ft)**	**bT** (K)	
10000			nrpts		

where Δr_c is the change in radius due to creep closure, R the initial cavern radius, R_g the gas constant, T is temperature, and At is the time step. The elastic response (time-independent) equation for cylindrical geometry[2] is:

$$\Delta r_e = - \left(\frac{\Delta p \; r}{E_m}\right)\left(1 - \nu\right) \qquad (2)$$

where Are is the change in radius due to the cavern's elastic response, Ap the change in pressure, r the radius, Em the modulus of elasticity, and ν Poisson's ratio. Recasting these equations in terms of the **"sprcreep.1"** input parameters is done as follows:

Variable Name From Equation (1) or (2)	**"sprcreep.1"** Input File Variable Name
R	Rft
A	**A_yr**
At	deltat
E/R_g	E
μ	mu
Klitho	Klitho
Em	Elast
ν	nu

The input parameters **"mT"** and **"bT"** are the constants which control the model's linear temperature profile according to the equation,

$$T = m_T \; z + b_T \qquad (3)$$

The final parameter in the **"sprcreep.1"** input file, "nrpts", is used to define the size of two arrays which are used to store the radius and pressure as functions of depth, r(z) and P(z). The minimum number of points necessary to insure accuracy for a 2000 ft cavern was found to be ten thousandl. The input parameters in **"sprcreep.1"** are listed, defined, and dimensionalized in Appendix B.

"sprcreep.9"

If the input parameter **"LPtConst"** is true, the program will run at constant **wellhead** pressure, removing brine and oil as defined by the input parameters contained in **"sprcreep.1"**, as discussed above. If "LPtConst" is false, the program needs to know when and how to remove brine or oil. The brine and oil removal for a nonconstant P_w run can be input to the program one of two ways, depending on whether the program is running interactively. If the program is running interactively, "Lterm" must be true, and the program will prompt the user for the necessary information. If the user wishes the program to run automatically,

without having P_w constant, **"Lterm"** must be false and the information that would have been input at the terminal must be supplied by an additional input file **"sprcreep.9".** Thus, if the program is running neither interactively (i.e. the program is running with "Lterm" false) nor at constant P_w (i.e. the program is running with **"LPtConst"** false), the program will require an additional input file. In either case, the brine and oil removal amounts specified by the user should be per time step. Examples of all modes of operation are included below.

"sprcreep.7"

As discussed above, the program can be run from a previous r(z) configuration. This simply requires the previous output of P_w, zi, and r(z) at the end of the last run (contained in the **"sprcreep.8"** output file as discussed below) to be used as input for the current run. This is accomplished by copying the appropriate **"sprcreep.8"** file to the name **"sprcreep.7"** and setting "nrpts" to 0 in the **"sprcreep.1"** input file. This signals the program to look for **"sprcreep.7"** and use its contents for inital **wellhead** pressure, initial interface depth, and initial r(z). Operated in this mode, the values of initial **wellhead** pressure and interface depth from **"sprcreep.7"** take priority over the values in **"sprcreep.1".** It is the user's responsibility to ensure that the other variables in **"sprcreep.1"** are appropriate for the new run. An example of this operation is discussed in the text that follows.

Output Files

"sprcreep.2"

When the program begins, it immediately records the **"sprcreep.1"** input variables and the initial masses of oil and brine in the **"sprcreep.2"** output file. The **"sprcreep.2"** output file is then used to record the time, **wellhead** pressure, interface depth, total volume, masses of brine and oil, and volume of brine and oil removed, at each time step. When the program finishes, the barrels of brine and oil removed are **totalled** and included at the end of this file. The **"sprcreep.2"** file is therefore best used to monitor the evolution of **wellhead** pressure, interface depth, and volume loss with time. An example of this output file for a constant **wellhead** pressure calculation has been included in Fig. 2.

"sprcreep. 3"

This file is used primarily to record the cavern radius profile, r(z), at the beginning and end of the calculations as well as at "tdump" time intervals, in years, in between. When an r(z) profile is dumped to this file, other parameters are also recorded, such as the pressure and temperature profiles with depth, **P(z)** and T(z). In addition, several variables are included in the far right column. The identity of the first two of these variables, **"Pw"** and "zi" are obvious, **wellhead** pressure and interface depth. The next three variables are also self explanatory: the mass of brine, the volume of brine removed (barrels) at the current time step, and the cumulative amount of brine removed (barrels) up to and including the current time. Three similar parameters for the oil follow. This list of parameters concludes with the volumes

Figure 2

"sprcreep.2" Output File for
a Run at Constant P_w

Pw	600.00000	Rft	100.00000		
zt	2000.00000	**zb**	4000.00000	zio	3750.00000
deltat	0.05000	tquit	1.00000	tdump	0.50000
DtMb	0.10000	**bcleak**	0.35000	ecleak	0.85000
LPtConst	T	Lterm	F		
Bcremove	150.00000	spgr	0.87600		
Klitho	1.0000	E	12581.78000	mu	**0.12319E+07**
A_yr	**0.24309E+30**				
Elast	**0.41660E+07**	nu	0.30000		
mT	**0.77780E-02**	**bT**	299.44000		
nrpts	10000				

Mass (t = 0) of Crude and Brine in reduced units

Mc @ t = 0 (in units of **10******6** kg) 1332.489
Mb @ t = 0 (in units of **10******6** kg) 265.106

time (yrs)	Pw (psia)	z1 (ft)	Vtot (10^6 ft^3)	Mb -(10^6 kg)-	Mc -(10^6 kg)-	Brine Removed (bbl)	Crude Removed (bbl)
0.000	599.999	3750.000	62.830	265.11	1332.49	0.0	0.0
0.050	608.826	3749.967	62.827	265.11	1332.49	0.0	0.0
0.100	599.999	3750.075	62.825	264.94	1332.49	875.0	0.0
0.150	608.825	3750.042	62.822	264.94	1332.49	0.0	0.0
0.200	599.999	3750.151	62.820	264.77	1332.49	874.9	0.0
0.250	608.825	3750.118	62.817	264.77	1332.49	0.0	0.0
0.300	599.999	3750.226	62.815	264.61	1332.49	874.8	0.0
0.350	605.780	3750.190	62.812	264.61	1332.47	0.0	150.0
0.400	599.999	3750.247	62.810	264.50	1332.45	573.0	150.0
0.450	605.779	3750.211	62.807	264.50	1332.43	0.0	150.0
0.500	599.999	3750.268	62.805	264.39	1332.41	572.9	150.0
0.550	605.778	3750.232	62.802	264.39	1332.38	0.0	150.0
0.600	599.999	3750.289	62.800	264.28	1332.36	572.8	150.0
0.650	605.778	3750.253	62.797	264.28	1332.34	0.0	150.0
0.700	599.999	3750.310	62.795	264.17	1332.32	572.8	150.0
0.750	605.777	3750.273	62.793	264.17	1332.30	0.0	150.0
0.800	599.999	3750.330	62.790	264.06	1332.28	572.7	150.0
0.850	608.823	3750.297	62.788	264.06	1332.28	0.0	0.0
0.900	599.999	3750.406	62.785	263.89	1332.28	874.3	0.0
0.950	608.822	3750.373	62.783	263.89	1332.28	0.0	0.0
1.000	599.999	3750.482	62.780	263.73	1332.28	874.2	0.0
						7237.43	1500.00

of the space occupied by the brine, oil, and the total cavern volume, all in cubic ft. An example of the **"sprcreep.3"** output file for the program running in constant **wellhead** pressure mode has been included in Fig. 3.

"sprcreep.8"

In addition to writing the r(z) profile to **"sprcreep.3"** at "tdump" intervals, the program records the pressure at the top of the cavern, the interface depth, the number of points in the r(z) array, and the r(z) array itself in the output file **"sprcreep.8".** This feature of the program allows it to be easily restarted in the event of a hardware failure, saving both real and CPU time. Because this file is intended to be used as an input configuration to restart the program (i.e. as the input file **"sprcreep.7")** the information contained in this file remains nondimensional, in the format used in the program itself. The program's nondimensionalization is documented with source code comment statements (Appendix A) as well as in Appendix B. Furthermore, in the event of a program restart, only the most recent r(z) written to **"sprcreep.8"** is required, thus when this information is written to the file, it overwrites the previously recorded information. This is accomplished by a FORTRAN REWIND statement after each r(z) dump to **"sprcreep.8".** An example of this file has been include in Fig. 4, with only the first 10 points of the r(z) array.

EXAMPLES

This creep closure model for SPR caverns allows several different types of analyses. For example the program can be operated in constant **wellhead** pressure mode as in the above figures. In this mode, "LPtConst" is chosen to be true, and the program removes amounts of brine calculated to be appropriate for maintaining constant **wellhead** pressure. The user defines how often brine is to be removed for this purpose (at intervals of **"DtMb"** years) and if, when, and how much oil should be leaked using the parameters "bcleak", "ecleak", and "Bcremove". In the above example, the user instructed the program to remove brine at every second time step. This was done by setting the brine removal interval **"DtMb"** equal to 0.1 years while the time step "deltat" was set to 0.05 years. In addition, the user set "bcleak" to 0.35 years, "ecleak" to 0.85 years, and "Bcremove" to 150 barrels, therefore the program leaked oil at the rate of 150 barrels per time step from 0.35 years to 0.85 years. As seen from Fig. 2, the program calculated the brine necessary to be removed every 0.1 years to bring the **wellhead** pressure back to its original value. This amount was dependent on if, and how much oil was leaking from the cavern. As seen in Fig. 3, the program recorded the radial profile at the beginning of the program (time = 0.0 years), and at "tdump" intervals (in years) thereafter.

If the program is operated in a nonconstant **wellhead** pressure mode **("LPtConst"** false), the user must decide if the program is to be run interactively. If so, "Lterm" is chosen to be true. An example of **"sprcreep.1"** with this selection is shown in Fig. 5.

Figure 3

"sprcreep.3" Output File for
a Run at Constant P_w

time (yrs) **0.00000**

z (ft)	**r (ft)**	**P (psia)**	**T (K)**
2000.00	100.00	1359.51	315.00
2040.00	100.00	1374.39	315.31
2080.00	100.00	1389.27	315.62
2120.00	100.00	1404.14	315.93
2160.00	100.00	1419.01	316.24
2200.00	100.00	1433.88	316.55
2240.00	100.00	1448.75	316.86
2280.00	100.00	1463.62	317.17
2320.00	100.00	1478.48	317.48
2360.00	100.00	1493.35	317.80
2400.00	100.00	1508.21	318.11
2440.00	100.00	1523.07	318.42
2480.00	100.00	1537.92	318.73
2520.00	100.00	1552.78	319.04
2560.00	100.00	1567.63	319.35
2600.00	100.00	1582.48	319.66
2640.00	100.00	1597.33	319.97
2680.00	100.00	1612.18	320.29
2720.00	100.00	1627.02	320.60
2760.00	100.00	1641.87	320.91
2800.00	100.00	1656.71	321.22
2840.00	100.00	1671.55	321.53
2880.00	100.00	1686.38	321.84
2920.00	100.00	1701.22	322.15
2960.00	100.00	1716.05	322.46
3000.00	100.00	1730.88	322.77
3040.00	100.00	1745.71	323.09
3080.00	100.00	1760.54	323.40
3120.00	100.00	1775.37	323.71
3160.00	100.00	1790.19	324.02
3200.00	100.00	1805.01	324.33
3240.00	100.00	1819.83	324.64
3280.00	100.00	1834.65	324.95
3320.00	100.00	1849.46	325.26
3360.00	100.00	1864.28	325.57
3400.00	100.00	1879.09	325.89
3440.00	100.00	1893.90	326.20
3480.00	100.00	1908.71	326.51
3520.00	100.00	1923.52	326.82
3560.00	100.00	1938.32	327.13

Pw	**(psia)**	599.999
zi	**(ft)**	3750.000
Mb	**(10^6** kg)	265.106
-Brine	**(bbl)**	0.002
-Brine tot	(bbl)	0.002
Mc	**(10^6** kg)	1332.489
-Crude	**(bbl)**	0.000
-Crude tot	(bbl)	0.000
vb	**(10^6 ft^3)**	7.853
vc	**(10^6 ft^3)**	54.977
Vtot	**(10^6 ft^3)**	62.830

3600.00	100.00	1953.12	327.44
3640.00	100.00	1967.92	327.75
3680.00	100.00	1982.72	328.06
3720.00	99.99	1997.52	328.37
3760.00	99.99	1997.52	328.69
3800.00	99.99	1997.52	329.00
3840.00	99.99	1997.52	329.31
3880.00	99.99	1997.52	329.62
3920.00	99.99	1997.52	329.93
3960.00	99.99	1997.52	330.24

time (yrs) 0.50000

z (ft)	r (ft)	P (psia)	T (K)
2000.00	100.00	1359.51	315.00
2040.00	100.00	1374.39	315.31
2080.00	100.00	1389.27	315.62
2120.00	100.00	1404.14	315.93
2160.00	100.00	1419.01	316.24
2200.00	100.00	1433.88	316.55
2240.00	100.00	1448.75	316.86
2280.00	100.00	1463.62	317.17
2320.00	100.00	1478.48	317.48
2360.00	100.00	1493.35	317.80
2400.00	100.00	1508.21	318.11
2440.00	100.00	1523.07	318.42
2480.00	100.00	1537.92	318.73
2520.00	100.00	1552.78	319.04
2560.00	100.00	1567.63	319.35
2600.00	100.00	1582.48	319.66
2640.00	100.00	1597.33	319.97
2680.00	100.00	1612.18	320.29
2720.00	100.00	1627.02	320.60
2760.00	100.00	1641.87	320.91
2800.00	100.00	1656.71	321.22
2840.00	100.00	1671.55	321.53
2880.00	100.00	1686.38	321.84
2920.00	100.00	1701.22	322.15
2960.00	100.00	1716.05	322.46
3000.00	99.99	1730.88	322.77
3040.00	99.99	1745.71	323.09
3080.00	99.99	1760.54	323.40
3120.00	99.99	1775.37	323.71
3160.00	99.99	1790.19	324.02
3200.00	99.99	1805.01	324.33
3240.00	99.99	1819.83	324.64
3280.00	99.98	1834.65	324.95
3320.00	99.98	1849.46	325.26
3360.00	99.98	1864.28	325.57
3400.00	99.98	1879.09	325.89
3440.00	99.97	1893.90	326.20
3480.00	99.97	1908.71	326.51
3520.00	99.97	1923.52	326.82

Pw	(psia)	599.999
zi	(ft)	3750.268
Mb	(10^6 kg)	264.387
-Brine	(bbl)	572.949
-Brine tot	(bbl)	3770.763
Mc	(10^6 kg)	1332.406
-Crude	(bbl)	150.000
-Crude tot	(bbl)	600.000
vb	(10^6 ft^3)	7.832
vc	(10^6 ft^3)	54.973
Vtot	(10^6 ft^3)	62.805

3560.00	99.96	1938.32	327.13
3600.00	99.96	1953.12	327.44
3640.00	99.95	1967.92	327.75
3680.00	99.95	1982.72	328.06
3720.00	99.94	1997.52	328.37
3760.00	99.93	1997.52	328.69
3800.00	99.93	1997.52	329.00
3840.00	99.92	1997.52	329.31
3880.00	99.91	1997.52	329.62
3920.00	99.90	1997.52	329.93
3960.00	99.89	1997.52	330.24

time (yrs) 1.00000

z (ft)	r (ft)	P (psia)	T (K)
2000.00	100.00	1359.51	315.00
2040.00	100.00	1374.39	315.31
2080.00	100.00	1389.27	315.62
2120.00	100.00	1404.14	315.93
2160.00	100.00	1419.01	316.24
2200.00	100.00	1433.88	316.55
2240.00	100.00	1448.75	316.86
2280.00	100.00	1463.62	317.17
2320.00	100.00	1478.48	317.48
2360.00	100.00	1493.35	317.80
2400.00	100.00	1508.21	318.11
2440.00	100.00	1523.07	318.42
2480.00	100.00	1537.92	318.73
2520.00	100.00	1552.78	319.04
2560.00	100.00	1567.63	319.35
2600.00	100.00	1582.48	319.66
2640.00	100.00	1597.33	319.97
2680.00	100.00	1612.18	320.29
2720.00	100.00	1627.02	320.60
2760.00	100.00	1641.87	320.91
2800.00	99.99	1656.71	321.22
2840.00	99.99	1671.55	321.53
2880.00	99.99	1686.38	321.84
2920.00	99.99	1701.22	322.15
2960.00	99.99	1716.05	322.46
3000.00	99.99	1730.88	322.77
3040.00	99.99	1745.71	323.09
3080.00	99.99	1760.54	323.40
3120.00	99.98	1775.37	323.71
3160.00	99.98	1790.19	324.02
3200.00	99.98	1805.01	324.33
3240.00	99.97	1819.83	324.64
3280.00	99.97	1834.65	324.95
3320.00	99.97	1849.46	325.26
3360.00	99.96	1864.28	325.57
3400.00	99.96	1879.09	325.89
3440.00	99.95	1893.90	326.20
3480.00	99.94	1908.71	326.51

Pw	(psia)	599.999
zi	(ft)	3750.482
Mb	(10^6 kg)	263.725
-Brine	(bbl)	874.165
-Brine tot	(bbl)	7237.432
Mc	(10^6 kg)	1332.280
-Crude	(bbl)	0.000
-Crude tot	(bbl)	1500.000
vb	(10^6 ft^3)	7.812
vc	(10^6 ft^3)	54.968
Vtot	(10^6 ft^3)	62.780

3520.00	99.94	1923.52	326.82
3560.00	99.93	1938.32	327.13
3600.00	99.92	1953.12	327.44
3640.00	99.91	1967.92	327.75
3680.00	99.90	1982.72	328.06
3720.00	99.88	1997.52	328.37
3760.00	99.87	1997.52	328.69
3800.00	99.86	1997.52	329.00
3840.00	99.84	1997.52	329.31
3880.00	99.83	1997.52	329.62
3920.00	99.81	1997.52	329.93
3960.00	99.79	1997.52	330.24

Figure 4

A Partial Listing of the
"sprcreep.8" Output File for
a Run at Constant P_w

```
4.7293006689549271E-03   37.50481674573856
     10000
0.9999990339360508
0.9999990326974712
0.9999990314575117
0.9999990302161711
0.9999990289734481
0.9999990277293415
0.9999990264838496
0.9999990252369714
0.9999990239887055
0.9999990227390507
```

Figure 5

"sprcreep.1" Input File for
an Interactive Run at Nonconstant P_w

600	100.0		**Pw (psia)**	Rft (ft)	
2000	4000	3750	zt (ft)	zb (ft)	**zi0** (ft)
0.05	1.0	0.50	deltat (yr)	**tquit (yr)**	**tdump (yr)**
0.10	0.35	0.85	**DtMb (yr)**	**bcleak** (yr)	ecleak (yr)
F	T		**LPtCONST**	Lterm	
150.0	0.876		Bcremove (bbl/deltat)		**spgr**
1.0	12581.78	**1.23188e6**	Klitho (psia/ft)	**E (K)**	mu (psia)
2.4309e+29			**A_yr (1/yr)**		
4.166e6	0.30		Elast (psia)	**nu**	
7.778e-3	299.44		**mT** (K/ft)	**bT** (K)	
10000			nrpts		

Figure 6

Terminal Log for
an Interactive Run at Nonconstant P_w

```
$ run creep13

  time      Pw       zi       Vtot      Mb     Mc        Brine      Crude
  (yrs)   (psia)    (ft)       6   3        6          Removed    Removed
                             (10  ft )  -(10 kg)-        (bbl)      (bbl)

  0.000   600.000 3750.000    62.830 265.11 1332.49         0.0        0.0

***      The current time is . . .      ***

                0.050

 *** Number of barrels brine removed & ***
 *** Number of barrels crude leaked  ? ***
 *** (enter a negative number for      ***
 ***    either to terminate this run)  ***

0.0
0.0

 *** Enter the time intervals on brine ***
 ***       and crude removals (yrs)     ***

0.05
0.05

 *** Enter the last time at which to   ***
 *** remove these amounts of brine and ***
 ***            crude (yrs)            ***
```

```
0.20
  0.050   608.826 3749.967   62.827 265.11 1332.49        0.0         0.0
  0.100   617.403 3749.934   62.825 265.11 1332.49        0.0         0.0
  0.150   625.743 3749.903   62.823 265.11 1332.49        0.0         0.0
  0.200   633.860 3749.872   62.821 265.11 1332.49        0.0         0.0


***      The current time is . . .      ***

                0.250


 *** Number of barrels brine removed & ***
 *** Number of barrels crude leaked  ? ***
 *** (enter a negative number for      ***
 ***    either to terminate this run)  ***


100.0
200.0


 *** Enter the time intervals on brine ***
 ***      and crude removals (yrs)     ***


0.10
0.05


 *** Enter the last time at which to   ***
 *** remove these amounts of brine and ***
 ***            crude (yrs)            ***


0.45
  0.250   635.700 3749.854   62.819 265.09 1332.46      100.0       200.0
  0.300   639.485 3749.820   62.817 265.09 1332.43        0.0       200.0
  0.350   641.178 3749.803   62.814 265.07 1332.41      100.0       200.0
  0.400   644.820 3749.770   62.812 265.07 1332.38        0.0       200.0
  0.450   646.373 3749.753   62.810 265.05 1332.35      100.0       200.0


***      The current time is . . .      ***

                0.500


 *** Number of barrels brine removed & ***
 *** Number of barrels crude leaked  ? ***
 *** (enter a negative number for      ***
 ***    either to terminate this run)  ***
```

```
50.0
0.0


 *** Enter the time intervals on brine ***
 ***      and crude removals (yrs)     ***


0.05
0.05


 *** Enter the last time at which to   ***
 *** remove these amounts of brine and ***
 ***            crude (yrs)            ***


1.5
  0.500   652.957   3749.732   62.808  265.04  1332.35        50.0          0.0
  0.550   659.375   3749.712   62.806  265.03  1332.35        50.0          0.0
  0.600   665.635   3749.692   62.805  265.02  1332.35        50.0          0.0
  0.650   671.743   3749.673   62.803  265.01  1332.35        50.0          0.0
  0.700   677.705   3749.654   62.801  265.00  1332.35        50.0          0.0
  0.750   683.529   3749.636   62.799  264.99  1332.35        50.0          0.0
  0.800   689.218   3749.618   62.797  264.98  1332.35        50.0          0.0
  0.850   694.779   3749.600   62.796  264.97  1332.35        50.0          0.0
  0.900   700.216   3749.583   62.794  264.96  1332.35        50.0          0.0
  0.950   705.533   3749.567   62.792  264.95  1332.35        50.0          0.0
  1.000   710.735   3749.551   62.791  264.94  1332.35        50.0          0.0
FORTRAN  STOP
```

Figure 7

"sprcreep.2" Output File for
an Interactive Run at Nonconstant P_w

Pw	600.00000	Rft	100.00000				
zt	2000.00000	zb	4000.00000	zi0	3750.00000		
deltat	0.05000	tquit	1.00000	tdump	0.50000		
DtMb	0.10000	bcleak	0.35000	ecleak	0.85000		
LPtConst	F	Lterm	T				
Bcremove	150.00000	spgr	0.87600				
Klitho	1.0000	E	12581.78000	mu	0.12319E+07		
A_yr	0.24309E+30						
Elast	0.41660E+07	nu	0.30000				
mT	0.77780E-02	bT	299.44000				
nrpts	10000						

Mass (t = 0) of Crude and Brine in reduced units

Mc @ t = 0 (in units of 10**6 kg) 1332.489
Mb @ t = 0 (in units of 10**6 kg) 265.106

time (yrs)	Pw (psia)	zi (ft)	Vtot (10^6 ft^3)	Mb -(10^6 kg)-	Mc	Brine Removed (bbl)	Crude Removed (bbl)
0.000	600.000	3750.000	62.830	265.11	1332.49	0.0	0.0
0.050	608.826	3749.967	62.827	265.11	1332.49	0.0	0.0
0.100	617.403	3749.934	62.825	265.11	1332.49	0.0	0.0
0.150	625.743	3749.903	62.823	265.11	1332.49	0.0	0.0
0.200	633.860	3749.872	62.821	265.11	1332.49	0.0	0.0
0.250	635.700	3749.854	62.819	265.09	1332.46	100.0	200.0
0.300	639.485	3749.820	62.817	265.09	1332.43	0.0	200.0
0.350	641.178	3749.803	62.814	265.07	1332.41	100.0	200.0
0,400	644.820	3749.770	62.812	265.07	1332.38	0.0	200.0
0.450	646.373	3749.753	62.810	265.05	1332.35	100.0	200.0
0.500	652.957	3749.732	62.808	265.04	1332.35	50.0	0.0
0.550	659.375	3749.712	62.806	265.03	1332.35	50.0	0.0
0.600	665.635	3749.692	62.805	265.02	1332.35	50.0	0.0
0.650	671.743	3749.673	62.803	265.01	1332.35	50.0	0.0
0.700	677.705	3749.654	62.801	265.00	1332.35	50.0	0.0
0.750	683.529	3749.636	62.799	264.99	1332.35	50.0	0.0
0.800	689.218	3749.618	62.797	264.98	1332.35	50.0	0.0
0.850	694.779	3749.600	62.796	264.97	1332.35	50.0	0.0
0.900	700.216	3749.583	62.794	264.96	1332.35	50.0	0.0
0.950	705.533	3749.567	62.792	264.95	1332.35	50.0	0.0
1.000	710.735	3749.551	62.791	264.94	1332.35	50.0	0.0
						850.00	1000.00

Figure 8

"sprcreep.1" Input File for
a Noninteractive Run at Nonconstant P_w

600	100.0		**Pw (psia)**	Rft (ft)	
2000	4000	3750	zt (ft)	zb (ft)	zio **(ft)**
0.05	1.0	0.50	deltat (yr)	tquit (yr)	**tdump (yr)**
0.10	0.35	0.85	**DtMb (yr)**	**bcleak** (yr)	ecleak (yr)
F	F		**LPtCONST**	Lterm	
150.0	0.876		Bcremove (bbl/deltat)		**spgr**
1.0	12581.78	**1.23188e6**	Klitho (psia/ft)	**E (K)**	mu (psia)
2.4309e+29			**A_yr (1/yr)**		
4.166e6	0.30		Elast (psia)	**nu**	
7.778e-3	299.44		**mT (K/ft)**	**bT** (K)	
10000			nrpts		

Figure 9

"sprcreep.9" Input File for
a Noninteractive Run at Nonconstant P_w

```
0.250   100.0       200.0
0.300     0.0       200.0
0.350   100.0       200.0
0.400     0.0       200.0
0.450   100.0       200.0
0.500    50.0         0.0
0.550    50.0         0.0
0.600    50.0         0.0
0.650    50.0         0.0
0.700    50.0         0.0
0.750    50.0         0.0
0.800    50.0         0.0
0.850    50.0         0.0
0.900    50.0         0.0
0.950    50.0         0.0
1.000    50.0         0.0
```

In this mode, the program first records the initial information in **"sprcreep.2"** and **"sprcreep.3"** and then prompts the user for brine and oil removal information. The program does this by announcing the current time, then asking for the number of barrels of brine and oil to be removed, the time steps for these removals, and the ending time for this information. The program then **calulates** the evolution of the system for these conditions and prompts the user for the next round of information when the end time is reached. An example of the terminal output when the program is run in this mode on a Digital Equipment Corporation VAX computer has been included in Fig. 6, with the **"sprcreep.2"** file included in Fig. 7. Notice that the program stops at the **"tquit"** specified in the **"sprcreep.1"** input file--the user interacting with the program cannot override this parameter from the terminal.

If the user chooses to run the program automatically, the information that the user would have provided the program interactively must be contained in an input file **"sprcreep.9".** For example, if the above nonconstant Pw run were to be done automatically, **"sprcreep.1"** would need to have "Lterm" set to false as in Fig. 8. In addition, **"sprcreep.9"** in the form of Fig. 9 would need to be available to the program. The **"sprcreep.9"** file, as seen in Fig. 9, contains all time steps, in the first column, at which nonconstant amounts of brine and/or oil are to be removed. These amounts are contained in the second and third columns respectively, and are in barrels per time step.

SUMMARY

With this simplified creep closure computer program, the user can model several different phenomena which occur in SPR caverns. The program has been designed to be easy to use and to provide information in much the same format as would be expected from a real SPR cavern. In particular, the user can simulate normal cavern operational methods by selecting the constant **wellhead** pressure option and setting **"DtMb,"** the interval at which brine is removed. By operating the program interactively, the user can quickly learn how the program operates while modelling short segments of cavern operation. Finally, through the use of the **"sprcreep.9"** input file, this program can by used to investigate creep closure with actual operational parameters over extended periods of time.

REFERENCES

1. G. S. Heffelfinger, 'Creep Closure of Salt Caverns in the Strategic Petroleum Reserve', Sandia Laboratories Report SAND 90-2614 (1991).

2. R. J. Roark, Formulas for Stress and Strain, 3rd Edition, McGraw-Hill, New York (1954).

Appendix A

Simplified Creep Model FORTRAN Source Code

```
*  creep.for
* March 27, 1991
*
* This version was documented for the SAND report user's guide.
* Microsoft (MS) Fortran compatability revision by P.S. Kuhlman 2/11/91.
* To run under MS Fortran, simply remove the C's in the lines
*  commented with C's.  Note that instead of using DO WHILE statements,
*  IF (...) GOTO's have been used.  While this is not according to strict
* Fortran 77 structured programming guidelines, it is compatible with MS
*  Fortran.
*
*  Subroutines and Functions:
*
*  subroutine  start.................This routine reads in the input
*                                    parameters,  sets-up other constants,
*                                    and initializes the arrays for the
*                                    numerical  integration.
*
*  function  MassCrude(z).........  This  function subroutine calculates
*                                    the differential mass of the crude,
*                                    i.e. when integrated, it yields the
*                                    mass of the crude.
*
*  function  MassBrine(z).........This function subroutine calculates
*                                    the differential mass of the brine,
*                                    i.e.  when integrated, it yields the
*                                    mass of the brine.
*
*  function VolCrude(z).............This  function subroutine calculates
*                                    the differential volume of the crude.
*                                    i.e.  when integrated, it yields the
*                                    volume of the crude.
*
*  function  VolBrine(z).............This function subroutine calculates
*                                    the differential volume of the brine,
*                                    i.e.  when integrated, it yields the
*                                    volume of the brine.
*
* function Pcrude(z)...............This function returns the pressure in
*                                    the cavern in the crude oil region. It
*                                    requires an integration and therefore a
*                                    second routine,  PcInteg is called as an
*                                    argument of the integration routine,
*                                    qgaus2.
*
* function PcInteg(z).............This function,  when integrated, provides
```

```
*                                    a necessary piece of the equation to
*                                    calculate Pcrude, and is called by the
*                                    function subroutine Pcrude.
*
*  function Pbrine(z).....This function returns the pressure in the
*                                    cavern in the brine region.
*
*  function radius(z,Pcavern).....The function 'radius' returns a value
*                                    for r(z) at the current time. This
*                                    is done by first "looking up" the value
*                                    of r(z) at the previous time step and
*                                    then, by using the creep equation to
*                                    calculate the change in r at that z,
*                                    calculating a new r(z).  Pcavern is the
*                                    pressure in the cavern at z, supplied as
*                                    either "Pcrude" or "Pbrine".
*
*  subroutine output..............This subroutine, called after every
*                                    time step, records the calculated
*                                    information.
*
*  subroutine  iterate(Mbnow,Mcnow) ..This subroutine performs the iteration
*                                    for "zi" and "Pt" using the mass balance
*                                    equations for brine and crude.
*
*  subroutine qgaus(func,a,b,ss)....This is the first of two subroutines
*                                    used to perform the numerical integrations
*                                    required.
*
*  subroutine qgaus2(func,a,b,ss)...This is the second of two subroutines
*                                    used to perform the numerical integrations
*                                    required.
*
* First, declare the variables in each common block. This is done
*  in similar fashion at the beginning of each subroutine.
*
*     implicit none
C $NOTRUNCATE
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8 Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
```

```
      real*8 CT,CT2,CT3
      real*8  alphac,alphab
*
      real*8 xinteg,winteg
      dimension xinteg(5),winteg(5)
*
      real*8 Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 phi,bigI,bigII,bigIII
*
*  Next, declare the variables used in the main program.
*
      integer i
      integer  icountm
      real*8 DtMc
      real*8 itMbremove,itMcremove
      real*8  ittime,itquit
*
      real*8 Mc,Mb,Vc,Vb
      real*8 Mbt0,Mct0
      real*8 z
*
      real*8 Mbnow,Mcnow
      real*8 Mbnowp,Mbnowm
      real*8 Mbremovep,Mbremovem
      real*8 Ptp,Ptm
      real*8  delta,slope,error
*
*  Declare and externalize the function subroutines.
*
      real*8  radius
      real*8  Pbrine,Pcrude
      real*8  MassCrude,MassBrine
      real*8  VolCrude,VolBrine
*
      external MassCrude
      external MassBrine
      external VolCrude
      external VolBrine
*
*  Set-up all common blocks used in the program.  The subroutines
*  include only the common blocks necessary for the execution of
*  that subroutine,  but the main program contains a copy of each
*  common block used in the program.
*
       common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                   DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                   Bcremove,spgr,Klitho,E,mu,A_yr,
     &                   Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                   Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                   CT,CT2,CT3,alphac,alphab
       common/integ/     xinteg,winteg
       common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
```

```
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
      common/brine/     phi,bigI,bigII,bigIII
      common/outputc/   Mc,Mb,Vc,Vb
*
* The following open statements (commented for VAX), facilitate the
* input/output of the parameters in the model.  "Creep.1" contains model
* parameters.  "Creep.2" is an output file which records the parameters
*  used in the model (from "creep.1") and writes wellhead pressure,
*  "Pw", interface depth, "zi", and other parameters at each time step.  This
* information (each time step) is also written to channel 6 (to terminal)
* for interactive use.  "Creep.3" is also an output data file, however,
* it contains information recorded at "tdump" time intervals.  ("tdump"
* is an input variable) This information is largely pressure,
*  temperature, and radius as functions of depth.  If the initial
* cavern geometry is not cylindrical, r(z) is read from the "creep.7"
*  input data file.  "Creep.8" is an output data file which is used to
* store r(z) at regulated time intervals ("tdump"), thus this file can
* be used to restart the model (say, if the program crashes) by reading it
*  into channel 7 as "creep.7".  In addition, the "creep.8" output
* file can be used as an initial r(z) to start a second calculation with
* different model parameters. Finally, "creep.9" is an input file to be used
*  to supply brine and crude removal parameters if the program is not run
* interactively.
*
*      open(unit=1,status='unknown',file='sprcreep.1')
*      open(unit=2,status='unknown',file='sprcreep.2')
*      open(unit=3,status='unknown',file='sprcreep.3')
*      open(unit=7,status='unknown',file='sprcreep.7')
*      open(unit=8,status='unknown',file='sprcreep.8')
*      open(unit=9,status='unknown',file='sprcreep.9')
*
      call start
*
*
* Before calling ANY routine which requires either "Pcrude"
*  or "Pbrine", "Pt", "zi", and "Pi" must be current.  For example, note that
*  several places in this program "Pi" is calculated just before calling
*  "qgaus" or "qgaus2", routines which are used to integrate function
*  subroutines in order to calculate the mass of brine or crude.  This is
* because the function subroutines that are integrated require "Pt", "zi",
*  and/or "Pi".  Perhaps the proper (structured) way of programing this would
* have been to pass these variables through the subroutine calls. However the
*  "qgaus" integration routines, taken from "Numerical Recipes", are written
*  such that the integrated function is a function of a single variable
* (i.e. "MassBrine" = f(z) not f("zi","Pi","z")). Thus the variables, "Pt",
*  "zi" and "Pi", are passed via common block, and therefore must be current
* before calling any integration routine(s) which utilize these parameters.
*
      zi = zio
      Pi = Pcrude(zi0)
*
* Determine initial masses and volumes of brine and crude, as well as
*  the original volume of the cavern itself.
*
```

```
      call qgaus(MassCrude,zt,zi,Mct0)
      call qgaus(MassBrine,zi,zb,Mbt0)
      call qgaus(VolCrude,zt,zi,Vc)
      call  qgaus(VolBrine,zi,zb,Vb)
*
* Record the calculated data.
*
      write(2,*)' '
      write(2,*)'Mass (t = 0) of Crude and Brine in reduced
     & units'
      write(2,*)' '
*
* Note on the following units:  multiplying a unitless parameter,
*  "Mct0" by g/cm**3 and ft**3 can then be multiplied by
* 10**6 cm**3/35.3145 ft**3 and then dividing by 1e+9 yielding
*  units of 1e+9 grams.
*
      write(2,1000)Mct0*Ccrg_cm3*Rft**3*1e+6/(35.3145*1e+9)
      write(2,1100)Mbt0*Ccrg_cm3*Rft**3*1e+6/(35.3145*1e+9)
1000 format(' ','Mc @ t = 0 (in units of 10**6 kg) ',f8.3)
1100 format(' ','Mb @ t = 0 (in units of 10**6 kg) ',f8.3)
*
*  Before stepping forward in time, initialize parameters used in
*  the time loop, and write out the headings to output channels 2 and 6.
*
      time = 0.0
*
      Mbnow = Mbt0
      Mbremove = 0.0
      Mbremovet = 0.0
*
*  Note that Mcremove is not set = 0.0 here.  This is because it has
*  already been initialized in the subroutine "start".
*
      Mcnow = Mct0
      Mcremovet = 0.0
*
      itMbremove = 0.0
      itMcremove = 0.0
*
      write(2,*)' '
      write(2,1200)
      write(2,1300)
      write(2,1400)
      write(2,*)' '
      write(6,*)' '
      write(6,*)' '
      write(6,*)' '
      write(6,*)' '
      write(6,*)' '
      write(6,*)' '
      write(6,*)' '
      write(6,1200)
      write(6,1300)
```

```
      write(6,1400)
      write(6,*)' '
1200  format(' ',2x,'time',6x,'Pw',7x,'zi',7x,'Vtot',6x,'Mb',5x,'Mc',8x,
     &'Brine',7x,'Crude')
1300  format(' ',2x,'(yrs)',3x,'(psia)',4x,'(ft)',5x,
     &'  6   3        6   ',8x,'Removed',5x,'Removed')
1400  format(' ',2x,'     ',3x,'      ',4x,'     ',4x,
     &'(10 ft )   -(10 kg)-',8x,'(bbl)',7x,'(bbl)')
*
* If operating at constant pressure ("LPtConst") then set "Mbremove" to
*  an initial estimate (chosen from experience).
*
      if (LPtConst) then
         Mbremove = DtMb*35000.0*Mbblb
         delta = 0.01*Mbremove
      else
         if (Lterm .eqv. .FALSE.) then
            read(9,*)ittime,itMbremove,itMcremove
         endif
      endif
*
* Begin the time loop.
*
1410  continue
         if (LPtConst) then
            if (time .ge. bcleak .and. time .le. ecleak) then
               Mcnow = Mcnow - Mcremove
               Mcremovet = Mcremovet + Mcremove
            endif
*
            if(time/DtMb - int(0.01 + time/DtMb) .le. 1.0e-2) then
               error = 1.0
               icountm = 0
*
* This loop is used to iterate around the mass of brine to be
*  removed so as to maintain constant pressure.
*
1420           continue
                  Mbremovep = Mbremove + delta/2.0
                  Mbremovem = Mbremove - delta/2.0
*
                  Mbnowp = Mbnow - Mbremovep
                  call  Iterate(Mbnowp,Mcnow)
                  Ptp = Pt
*
                  Mbnowm = Mbnow - Mbremovem
                  call  Iterate(Mbnowm,Mcnow)
                  Ptm = Pt
*
                  slope = (Ptp - Ptm)/(Mbremovep - Mbremovem)
                  Mbremove = (Pt0 - Ptp  +  slope*Mbremovep)/slope
                  error = abs(Pt0 - (Ptp + Ptm)/2.0)
*
                  icountm = icountm + 1
```

```
                  if (icountm .ge. 21) then
                     write(2,*)' ** ** **'
                     write(2,*)'20 iterations on Mbremove have occured'
                     write(2,*)'therefore the loop has been bypassed'
                     write(2,*)' ** ** **'
                     error = 0.0
                     Mbremove = (Mbremovep + Mbremovem)/2.0
                  endif
               if (error .gt.  l.Oe-9) goto 1420
*
               if (Mbremove .lt. 0.0) then
                  Mbremove = 0.0
               end if
            else
               Mbremove = 0.0
            endif
            Mbnow = Mbnow - Mbremove
            Mbremovet = Mbremovet + Mbremove
         else
            Mcnow = Mcnow - Mcremove
            Mcremovet = Mcremovet + Mcremove
*
            Mbnow = Mbnow - Mbremove
            Mbremovet = Mbremovet + Mbremove
         endif
*
         if (Mbnow .le. O.O)then
            write(6,*)'Error:  the amount of brine removed has'
            write(6,*)'exceeded the initial amount of brine'
            stop
         endif
*
         call  Iterate(Mbnow,Mcnow)
*
         call  qgaus(MassCrude,zt,zi,Mc)
         call  qgaus(MassBrine,zi,zb,Mb)
*
         call  qgaus(VolCrude,zt,zi,Vc)
         call  qgaus(VolBrine,zi,zb,Vb)
*
         call output
*
* Update the "Psave" and "rsave" arrays which contain "Psave" and
*  r(z) at the current time.
*
         z = zt
         i - l
1430     continue
            rsave(i) =  radius(z,Pcrude(z))
            Psave(i) = Pcrude(z)
            z = z + deltazr
            i = i + 1
         if (z .lt. zi) got0 1430
*
```

```
          z = z + deltazr
          i = i + l
*
1440      continue
             rsave(i) =  radius(z,Pbrine(z))
             Psave(i) = Pbrine(z)
             z = z + deltazr
             i = i + 1
          if (z .le. zb) goto 1440
*
          rsave(i) = rsave(nrpts)
          rsave(i+l) = rsave(nrpts)
          Psave(i) = Psave(nrpts)
          Psave(i+l) = Psave(nrpts)
*
* This next set of if structures enables the program to ask
*  the user how to proceed if the program is used interactively.
*
          if (LPtConst .eqv. .FALSE.) then
             if (Lterm) then
                if (abs(time-itquit) .le. l.Oe-06) then
                   write(6,*)' '
                   write(6,*)' '
                   write(6,1500)
                   write(6,*)' '
                   write(6,*)' '
                   write(6,1600)time + deltat
1500               format(' ***      The current time is . . .       ***')
1600               format('               ',f7.3)
*
                   write(6,*)' '
                   write(6,*)' '
                   write(6,*)' *** Number of barrels brine removed & ***'
                   write(6,*)' *** Number of barrels crude leaked  ? ***'
                   write(6,*)' *** (enter a negative number for      *** '
                   write(6,*)' ***    either to terminate this run)  *** '
                   write(6,*)' '
                   write(6,*)' '
                   read(5,*)itMbremove,itMcremove
*
                   itMbremove = itMbremove*Mbblb
                   itMcremove = itMcremove*Mbblc
*
                   if (itMbremove .lt. 0.0 .and. itMcremove .lt. 0.0)
     &              goto 1800
                   write(6,*)' '
                   write(6,*)' '
                   write(6,*)' *** Enter the time intervals on brine ***'
                   write(6,*)' ***      and crude removals (yrs)     ***'
                   write(6,*)' '
                   write(6,*)' '
                   read(5,*)DtMb,DtMc

                   write(6,*)' '
```

```
                  write(6,*)' '
                  write(6,*)' *** Enter the last time at which to   *** '
                  write(6,*)' *** remove these amounts of brine and ***'
                  write(6,*)' ***             crude (yrs)           ***'
                  write(6,*)' '
                  write(6,*)' '
                  read(5,*)itquit

                  ittime = time

                  Mbremove = itMbremove
                  Mcremove = itMcremove
               else
                  if(mod(sngl(time-ittime+1e-06),sngl(DtMb))
     &               .le. 1.0e-2)then
                     Mbremove = itMbremove
                  else
                     Mbremove = 0.0
                  endif
*
                  if(mod(sngl(time-ittime+1e-06),sngl(DtMc))
     &               .le. 1.0e-2)then
                     Mcremove = itMcremove
                  else
                     Mcremove = 0.0
                  endif
               endif
            else
               if (abs((time + deltat) - ittime) .le. 1.0e-06) then
                  Mbremove = itMbremove*Mbblb
                  Mcremove = itMcremove*Mbblc
                  read(9,*,end=1700)ittime,itMbremove,itMcremove
1700              continue
               else
                  Mbremove = 0.0
                  Mcremove = 0.0
               endif
            endif
         endif

         time = time + deltat
         if (sngl(time-deltat/8) .gt. sngl(tquit)) time = -100
         if (itMbremove .lt. 0.0) then
            if(Lterm) then
               time = -100.0
            else
               if (time .eq. ittime) time = -100.0
            endif
          endif
*
      if (time .gt. -1.0) got0 1410
*
1800  write(2,1900)
1900  format(' ',5(9x,2x),'----------------------')
```

```
      write(2,2000)Mbremovet/Mbblb,Mcremovet/Mbblc
2000  format(' ',49x,2(2x,f12.2))
*
* Finally, end t h program by writing out the final r(z) to channel 8
*
      rewind 8
      if (LPtConst) then
         write(8,*)Pt0,zi
      else
         write(8,*)Pt,zi
      endif
      write(8,*)nrpts
      do 2100 i = 1,nrpts+2
         write(8,*)rsave(i)
2100  continue
      stop
      end
* * *
      subroutine start
*
*  This routine reads in the input
*  parameters,  sets-up other constants,
*  and initializes the arrays for the
*  numerical  integration.
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8 Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
      real*8 CT,CT2,CT3
      real*8 alphac,alphab
*
      real*8 xinteg,winteg
      dimension xinteg(5),winteg(5)
*
      real*8 Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 phi,bigI,bigII,bigIII
      real*8 CbP_psia,CbT_K,CbT_K2,Cbt_K3,Cbrg_cm3
*
```

```
      integer i
      real*8 z,T
      real*8 Pcrude,Pbrine
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/integ/     xinteg,winteg
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
      common/brine/     phi,bigI,bigII,bigIII
*
* First, read in the constants from the channel 1 input file, then
* begin channel 2 output data file.
*
      read(l,*)Pw,Rft
      read(1,*)zt,zb,zi0
      read(l,*)deltat,tquit,tdump
      read(1,*)DtMb,bcleak,ecleak
      read(1,100)LPtConst,Lterm
100   format(2L11)
      read(l,*)Bcremove,spgr
      read(1,*)Klitho,E,mu
      read(1,*)A_yr
      read(l,*)Elast,nu
      read(l,*)mT,bT
      read(l,*)nrpts
*
      write(2,200)Pw,Rft
      write(2,300)zt,zb,zi0
      write(2,400)deltat,tquit,tdump
      write(2,500)DtMb,bcleak,ecleak
      write(2,600)LPtConst,Lterm
      write(2,700)Bcremove,spgr
      write(2,800)Klitho,E,mu
      write(2,900)A_yr
      write(2,1000)Elast,nu
      write(2,1100)mT,bT
      write(2,1200)nrpts
*
200   format(' ','Pw        ',f12.5,' Rft       ',f12.5)
300   format(' ','zt        ',f12.5,' zb        ',f12.5,
     &' zi0       ',f12.5)
400   format(' ','deltat    ',f12.5,' tquit     ',f12.5,
     &' tdump     ',f12.5)
500   format(' ','DtMb      ',f12.5,' bcleak    ',f12.5,
     &' ecleak    ',f12.5)
600   format(' ','LPtConst ',L12,' Lterm     ',L12)
700   format(' ','Bcremove ',f12.5,' spgr      ',f12.5)
800   format(' ','Klitho    ',g12.5,' E         ',f12.5,
     &' mu        ',g12.5)
```

```
900   format(' ','A_yr      ',g12.5)
1000  format(' ','Elast     ',g12.5,'  nu        ',f12.5)
1100  format(' ','mT        ',g12.5,'  bT        ',f12.5)
1200  format(' ','nrpts     ',i12)
*
* The program is entirely nondimensional, except for the variables which
* have dimensions of time.  Therefore the dimensional input parameters
*  must be converted to nondimensional form.  These first few variables are
*  the constants (from the crude equation of state) used to nondimensionalize
*  the rest of the program variables.  In particular, CcP_psia, the crude
*  equation of state pressure coefficient is evaluated at the temperature
*  at the top of the cavern.
*
      CcP_psia = (0.018*(mT*zt + bT) - 6.43*spgr + 3.6)*1e-6
      CcP_psia = CcP_psia/(1.0000945*spgr + 0.1695)
      CcT_K = 5.7627e-4/(1.0000945*spgr + 0.1695)
      Ccrg_cm3 = 0.98807*(1.0000945*spgr + 0.1695)
*
      CbP_psia = 1.722e-6
      CbT_K = 3.272e-3
      CbT_K2 = 8.933e-6
      CbT_K3 = -8.4868e-9
      Cbrg_cm3 = 2.0149
*
* The two parameters bigX and bigY are calculated from the dimensional
* values of mT and bT.  Once this is accomplished, all four parameters
*  can be nondimensionalized.
*
      bigX = 0.018*mT*1e-6/(1.0000945*spgr + 0.1695)
      bigY = (0.018*bT - 6.43*spgr + 3.6)*1e-6
      bigY - bigY/(1.0000945*spgr + 0.1695)
*
      Pw = Pw*CcP_psia
      zt = zt/Rft
      zb = zb/Rft
      zi0 - zi0/Rft
      klitho - klitho*CcP_psia*Rft
      E = E*CcT_K
      mu - mu*CcP_psia
      Elast - Elast*CcP_psia
*
      bigX - bigX*Rft/CcP_psia
      bigY = bigY/CcP_psia
      mT = mT*Rft*CcT_K
      bT = bT*CcT_K
*
* Initialize other nondimensional constants.
*
      Crho = Ccrg_cm3/Cbrg_cm3
      CP - CcP_psia/CbP_psia
      CT = CbT_K/CcT_K
      CT2 = CbT_K2/(CcT_K)**2
      CT3 = CbT_K3/(CcT_K)**3
*
```

```
* The following dimensionless parameters "alphab" and "alphac",
*  are defined to be:
*
* alphab = Cbrg_cm3*G*CbP_psia*Rft
*  alphac = Ccrg_cm3*G*CcP_psia*Rft
*
* where G is the gravitational constant.  For ease, G's units are adjusted
*  as follows:
*           m     1 kg     14.696 psia          l m          le6 cm**3
* G = 9.8 ----- * -----* ---------------- *-em------ * ----------
*          s**2   1000g   1.01325e5  k g     3.2808 ft      m**3
*                                   ------
*                                   m s**2
*
*              cm**3 psia
*  G = 0.43324 ----------
*                  ft
*
      alphab = Cbrg_cm3*CbP_psia*0.43324*Rft
      alphac = Ccrg_cm3*CcP_psia*0.43324*Rft
*
* The masses of a barrel of brine and crude, "Mbblb" and "Mbblc" are
*  calculated using 1.2 g/cm3 for the density of brine and the input
*  variable "spgr" for the specific gravity of crude and nondimensionalized
* by "Ccrg_cm3" and "Rft".  "Mbblb" is thus calculated:
*
*                g          gal    1 ft**3
* Mbblb = 1.2 --we-    * 42 --- * ----------
*             cm**3         bbl   7.4805 gal
*          --------------------------------
*                Ccrg_cm3 * Rft**3
*
*  and "Mbblc":
*
*                   1 g        gal    1 ft**3
* Mbblc = "spgr" * ----- * 42 --- * ----------
*                  cm**3       bbl   7.4805 gal
*          -----------------------------------
*                   Ccrg_cm3 * Rft**3
*
      Mbblb = (1.2*42.0/7.4805)/(Ccrg_cm3*Rft**3)
      Mbblc = (spgr*42.0/7.4805)/(Ccrg_cm3*Rft**3)
*
* Convert the dimensional Bcremove (barrels crude to be leaked per
*  time step) to nondimensional mass format so that it can be subtracted
* from the nondimensional mass of the crude in the cavern, Mcnow.
*
      if (LPtConst) then
         Mcremove = Bcremove*Mbblc
      else
         Mcremove = 0.0
      endif
*
* Calculate the nondimensional pressure at the top of the cavern, Pt.
```

```
*  Although all program input/output is in terms of the wellhead pressure,
*  Pw, throughout the program Pt is used.  Pt0 represents Pt at time
*  t = 0.  Also, initialize Pt by setting it equal to Pt0.
*
      Pt0 = Pw + zt*Rft*spgr*14.696*CcP_psia/33.9
      Pt = Pt0
*
*  The following two IF statements have been added to prevent
*  inappropriate inputs or combinations of inputs.
*
      if (LPtConst .and. Lterm) then
         write(6,*)' '
         write(6,*)'*********************************************'
         write(6,*)'This program is not designed to be run at'
         write(6,*)'constant wellhead pressure via terminal inputs'
         write(6,*)'therefore it is assumed that non-constant'
         write(6,*)'wellhead pressure is desired, thus'
         write(6,*)'LPtConstant has been set to .FALSE.'
         write(6,*)'and the program will be run from the terminal'
         write(6,*)'*********************************************'
         write(6,*)' '
         LPtConst = .FALSE.
      endif
*
*  If the program is running from terminal inputs, the user should
*  be able to remove crude at will.  The next if statements insures this.
*
      if (Lterm .eqv. .true.)bcleak = deltat
*
      if (deltat .eq. 0.0) then
         write(6,*)'Error:  This program requires a finite'
         write(6,*)'time step:  deltat (in the input file)'
         stop
      endif
*
*  Next, initialize the constants to be used in the brine equation of state.
*
      phi = -CT*bT + CT2*bT**2 + CT3*bT**3
      big1 = -CT*mT + 2.0*CT2*bT*mT + 3.0*CT3*mT*bT**2
      big11 = CT2*mT**2 + 3.0*CT3*bT*mT**2
      big111 = CT3*mT**3
*
*  Set-up the arrays for the numerical integration subroutines,
*  qgaus and qgaus2
*
      xinteg(1) = 0.1488743389
      xinteg(2) = 0.4333953941
      xinteg(3) = 0.6794095682
      xinteg(4) = 0.8650633666
      xinteg(5) = 0.9739065285
*
      winteg(1) = 0.2955242247
      winteg(2) = 0.2692667193
      winteg(3) = 0.2190863625
```

```
      winteg(4) = 0.1494513491
      winteg(5) = 0.0666713443
*
* Initialize r(z) by setting it equal to R (r nondim = 1.0) for
*  all z, or reading it from channel 7.
*
      if (nrpts .ne. 0) then
         do 2000 i = 1,nrpts + 2
            rsave(i) = 1.0
2000     continue
      else
         read(7,*)Pt0,zi0
         read(7,*)nrpts
         do 2100 i = 1,nrpts+2
            read(7,*)rsave(i)
2100     continue
      endif
*
* Set-up the spacing of the r(z) and Psave(z) arrays.
*
      deltazr = (zb - zt)/float(nrpts)
*
* Set-up the initial P(z) profile.
*
      z = zt
      i - l
*
1950  continue
         T = mT*z + bT
         Psave(i) = Pcrude(z)
         z = z + deltazr
         i = i + 1
      if (z .le. zi0) got0 1950
*
1960  continue
         T = mT*z + bT
         Psave(i) = Pbrine(z)
         z = z + deltazr
         i = i + 1
      if (z .le. zb) goto 1960
*
* Note that in the initialization of Psave and rsave, two "extra"
* points are initialized.  This is because the Newton-Gregory
*  table interpolation method uses two values beyond the requested
*  value.
*
      Psave(i) = Psave(nrpts)
      Psave(i+l) = Psave(nrpts)
*
      return
      end
* * *
      real*8 function MassCrude
*
```

```
* This function subroutine calculates
* the differential mass of the crude,
* i.e. when integrated, it yields the
* mass of the crude.
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8  Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
      real*8 CT,CT2,CT3
      real*8  alphac,alphab
*
      real*8 xinteg,winteg
      dimension xinteg(5),winteg(5)
*
      real*8  Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 T
      real*8 z
      real*8 rhoc,Pc
      real*8 pie
*
      real*8  radius,Pcrude
*
      external Pcrude
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/integ/     xinteg,winteg
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      pie = acos(-1.0)
*
      T = mT*z + bT
      PC = Pcrude(z)
```

```
      rhoc = 1 - T + Pc
*
      MassCrude = rhoc*pie*radius(z,Pc)**2
*
      return
      end
* * *
      real*8 function MassBrine
*
*  This function subroutine calculates
*  the differential mass of the brine,
*  i.e. when integrated, it yields the
*  mass of the brine.
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8 Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
      real*8 CT,CT2,CT3
      real*8 alphac,alphab
*
      real*8  Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8  phi,bigI,bigII,bigIII
*
      real*8 T
      real*8 z
      real*8 rhob,Pb
      real*8 pie
*
      real*8 radius,Pbrine
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     6                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
```

```
      common/brine/     phi,bigI,bigII,bigIII
*
      pie = acos(-1.0)
*
      T = mT*z + bT
      Pb = Pbrine(z)
      rhob = (1 + phi + bigI*z + bigII*z**2 + bigIII*z**3 + Pb/CP)/Crho
*
      MassBrine = rhob*pie*radius(z,Pb)**2
*
      return
      end
* * *
      real*8  function  VolCrude(z)
*
* This function subroutine calculates
*  the differential volume of the crude.
*  i.e. when integrated, it yields the
*  volume of the crude.
*
*     implicit  none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer  nrpts
*
      real*8 bigX,bigY
      real*8 Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
      real*8 CT,CT2,CT3
      real*8 alphac,alphab
*
      real*8 Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 z
      real*8 pie
*
      real*8 radius,Pcrude
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
```

```
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      pie = acos(-1.0)
*
      VolCrude = pie*radius(z,Pcrude(z))**2
*
      return
      end
* * *
      real*8 function VolBrine(z)
*
*  This function subroutine calculates
*  the differential volume of the brine,
*  i.e. when integrated, it yields the
*  volume of the brine.
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8  Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8  Crho,CP
      real*8 CT,CT2,CT3
      real*8  alphac,alphab
*
      real*8   Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 z
      real*8 pie
*
      real*8 radius,Pbrine
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
```

```
      pie = acos(-1.0)
*
      VolBrine = pie*radius(z,Pbrine(z))**2
*
      return
      end
* * *
      real*8 function Pcrude(z)
*
* This function returns the pressure in
*  the cavern in the crude oil region. It
*  requires an integration and therefore a
*  second routine, PcInteg is called as an
*  argument of the integration routine,
* qgaus2.
*
*     implicit none
      real*8 Pw,Rft
      real+8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8  Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8 Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
      real*8 CT,CT2,CT3
      real*8  alphac,alphab
*
      real*8 Pt,PtO,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 z
*
      real*8  PcInteg
      external PcInteg
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      call qgaus2(PcInteg,zt,z,Pcrude)
```

```
      Pcrude = (Pcrude +
     &      Pt*exp(-alphac*bigX*(zt**2)/2.0 - alphac*bigY*zt))/
     &      exp(-alphac*bigX*(z**2)/2.0 - alphac*bigY*z)
*
      return
      end
* * *
      real*8 function PcInteg(z)
*
*  This function, when integrated, provides
*  a necessary piece of the equation to
*  calculate Pcrude, and is called by the
*  function subroutine Pcrude.
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical  LPtConst,Lterm
      real*8  Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8  Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8  Crho,CP
      real*8 CT,CT2,CT3
      real*8  alphac,alphab
*
      real*8  Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 z
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     6                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      PcInteg = (alphac - alphac*bT - mT*alphac*z)*exp(
     &          -alphac*bigX*(z**2)/2.0 - alphac*bigY*z)
*
      return
      end
* * *
```

```
      real*8 function Pbrine(z)
*
*  This function returns the pressure in the
*  cavern in the brine region.
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8 Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
      real*8 CT,CT2,CT3
      real*8 alphac,alphab
*
      real*8 Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 phi,bigI,bigII,bigIII
*
      real*8 z,expon,termA,termB,termC
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
      common/brine/     phi,bigI,bigII,bigIII
*
      expon = exp(alphab*(z - zi))
      termA = CP*(1.0 + phi)*(expon - 1.0)
*
      termB =         bigIII*(zi**3*expon - z**3)/alphab
      termB = termB + (3.0*bigIII/alphab**2 + bigII/alphab)*
     &                ((zi**2)*expon - z**2)
      termB = termB + (6.0*bigIII/alphab**3 + 2.0*bigII/alphab**2 +
     &                 bigI/alphab)*((zi + 1.0/alphab)*expon -
     6                (z + 1.0/alphab))
      termB = alphab*CP*termB
*
      termC = Pi*expon
```

```
*
      Pbrine -termA+termB+termC
*
      return
      end
* * *
      real*8 function radius(z,Pcavern)
*
* The function 'radius' returns a value
* for r(z) at the current time. This
* is done by first "looking up" the value
*  of r(z) at the previous time step and
*  then, by using the creep equation to
*  calculate the change in r at that z,
* calculating a new r(z).  Pcavern is the
* pressure in the cavern at z, supplied as
*  either "Pcrude" or "Pbrine".
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8  Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8 Crho,CP
      real*8 CT,CT2,CT3
      real*8 alphac,alphab
*
      real*8  Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 T
      real*8 z
      real*8 Pcavern
*
* Declare the variables used in a Newton Gregory table interpolation.
*
      integer ifromz
      real*8 reali,zeta
      real*8 rnone,r1,r2,rtempo1,rtempo2,rold
      real*8 Pnone,P1,P2,Ptempo1,Ptempo2,Pold
*
      common/startups/ Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                 DtMb,bcleak,ecleak,LPtConst,Lterm,
```

```
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      T = mT*z + bT
*
*  The variable "ifromz" is the conversion from "z" (a real*8 variable)
*  to "i", the integer equivalent of "z", used with the array "rsave".
*
      reali = (z - zt)/deltazr + 1
      ifromz = int(reali)
      zeta = reali - float(ifromz)
*
      rnone = rsave(ifromz)
      Pnone = Psave(ifromz)
*
      rl = rsave(ifromz + 1)
      r2 = rsave(ifromz + 2)
      Pl = Psave(ifromz + 1)
      P2 = Psave(ifromz + 2)
*
      rtempol = rnone + (rl - rnone)*zeta
      rtempo2 = rl + (r2 - rl)*(zeta - 1.0)
      Ptempol = Pnone + (Pl - Pnone)*zeta
      Ptempo2 = Pl + (P2 - Pl)*(zeta - 1.0)
*
      rold = rtempol + (rtempo2 - rtempol)*zeta*0.5
      Pold = Ptempol + (Ptempo2 - Ptempol)*zeta*0.5
*
      radius = rold - deltat*A_yr*exp(-E/T)*
     &         (abs(Klitho*z - Pcavern)/mu)**5.5 -
     &         rold*(1.0 - nu)*(Pold - Pcavern)/Elast
*
      if (radius .lt. 0.0)radius = 0.0
*
      return
      end
* * *
      subroutine output
*
* This subroutine, called after every
*  time step, records the calculated
*  information.
*
*     implicit none
      real*8 Pw,Rft
      real*8 zt,zb,zi0
      real*8 deltat,tquit,tdump
      real*8 DtMb,bcleak,ecleak
      logical LPtConst,Lterm
      real*8 Bcremove,spgr
```

```
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8  Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8  Crho,CP
      real*8 CT,CT2,CT3
      real*8  alphac,alphab
*
      real*8 xinteg,winteg
      dimension xinteg(5),winteg(5)
*
      real*8  Pt,PtO,Pi,zi,time,rsave,Psave
      real*8   deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8 Mc,Mb,Vc,Vb
*
      real*8 r,T
      real*8 Pc,Pb
*
      real*8 Vtot
      real*8 dz,z
*
      integer i,npara
      parameter(npara = 11)
      real*8 para(npara)
      character*16  name(npara)
*
      real*8 radius
      real*8  Pcrude,Pbrine
*
      common/startups/  Pw,Rft,zt,zb,ziO,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     &                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     &                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/integ/     xinteg,winteg
      common/variables/ Pt,PtO,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
      common/outputc/   Mc,Mb,Vc,Vb
*
      Vtot = Vc + Vb
*
      para(1) = Pt/CcP_psia - zt*Rft*spgr*14.696/33.9
      para(2) = zi*Rft
      para(3) = Mb*Ccrg_cm3*Rft**3*1e+6/(35.3145*1e+9)
      para(4) = Mbremove/Mbblb
      para(5) = Mbremovet/Mbblb
      para(6) = Mc*Ccrg cm3*Rft**3*1e+6/(35.3145*1e+9)
```

```
*
*  The following if statements insure correct output regardless
*  of mode of program operation
*
      if (LPtConst) then
         if (time .ge. bcleak .and. time .le. ecleak) then
            para(7) = Mcremove/Mbblc
         else
            para(7) = 0.0
         endif
      else
            para(7) = Mcremove/Mbblc
      endif
*
      para(8) = Mcremovet/Mbblc
      para(9) = Vb*Rft**3/1e+6
      para(l0) = Vc*Rft**3/1e+6
      para(11) = Vtot*Rft**3/1e+6
*
      name(l) = 'Pw        (psia)'
      name(2) = 'zi          (ft)'
      name(3) = 'Mb     (10^6 kg)'
      name(4)  = '-Brine     (bbl)'
      name(5) = '-Brine tot (bbl)'
      name(6) = 'Mc     (10^6 kg)'
      name(7) = '-Crude     (bbl)'
      name(8) = '-Crude tot (bbl)'
      name(9) = 'Vb   (10^6 ft^3)'
      name(l0) = 'Vc   (10^6 ft^3)'
      name(11) = 'Vtot (10^6 ft^3)'
*
      if (mod(sngl(time)+1.E-6,sngl(tdump)) .le. l.Oe-2) then
*
         rewind 8
         if (LPtConst) then
            write(8,*)Pt0,zi
         else
            write(8,*)Pt,zi
         endif
         write(8,*)nrpts
         do 100 i = 1,nrpts+2
            write(8,*)rsave(i)
100      continue
*
         write(3,*)' '
         write(3,200)time
200      format(' ','time (yrs )  ',f12.5)
         write(3,*)' '
         write(3,300)
         write(3,400)
300      format(' ',' z      ',2x,' r     ',3x,' P',9x,'T')
400      format(' ',' (ft)    ',2x,' (ft) ',3x,'(psia)',5x,'(K)')
*
         dz = (zb - zt)/50.0
```

```
         z = zt
         i - l
*
450      continue
            T = mT*z + bT
            PC = Pcrude(z)
            r = radius(z,Pc)
*
            if (i .le. 11) then
               write(3,500)z*Rft,r*Rft,Pc/CcP_psia,T/CcT_K,
     &                     name(i),para(i)
500            format(' ',f7.2,3x,f6.2,3x,f7.2,3x,f6.2,3x,a16,2x,f12.3)
               i = i + l
            else
               write(3,500)z*Rft,r*Rft,Pc/CcP_psia,T/CcT_K
            endif
            z - z + d z
         if (z .le. zi) got0 450
*
550      continue
            T = mT*z + bT
            Pb = Pbrine(z)
            r = radius(z,Pb)
            write(3,500)z*Rft,r*Rft,Pc/CcP_psia,T/CcT_K
            z = z + dz
         if (z .le. zb) goto 550
       endif
*
       write(2,600)time,para(1),zi*Rft,Vtot*Rft**3/1e+6,
     &            Mb*Ccrg_cm3*Rft**3*1e+6/(35.3145*1e+9),
     &            Mc*Ccrg_cm3*Rft**3*1e+6/(35.3145*1e+9),
     &            Mbremove/Mbblb,para(7)
600    format(' ',f7.3,2x,f8.3,2x,f8.3,2x,f7.3,1x,f6.2,1x,f7.2,
     &2(1x,f11.1))
*
       write(6,600)time,para(1),zi*Rft,Vtot*Rft**3/1e+6,
     &            Mb*Ccrg_cm3*Rft**3*1e+6/(35.3145*1e+9),
     &            Mc*Ccrg_cm3*Rft**3*1e+6/(35.3145*1e+9),
     &            Mbremove/Mbblb,para(7)
*
       return
       end
* * *
       subroutine  iterate(Mbnow,Mcnow)
*
* This subroutine performs the iteration
* for "zi" and "Pt" using the mass balance
*  equations for brine and crude.
*
*      implicit none
       real*8 Pw,Rft
       real*8 zt,zb,zi0
       real*8 deltat,tquit,tdump
       real*8 DtMb,bcleak,ecleak
```

```
      logical LPtConst,Lterm
      real*8  Bcremove,spgr
      real*8 Klitho,E,mu
      real*8 A_yr
      real*8 Elast,nu
      real*8 mT,bT
      integer nrpts
*
      real*8 bigX,bigY
      real*8 Mbblb,Mbblc
      real*8 CcP_psia,CcT_K,Ccrg_cm3
      real*8  Crho,CP
      real*8 CT,CT2,CT3
      real*8  alphac,alphab
*
      real*8 xinteg,winteg
      dimension xinteg(5),winteg(5)
*
      real*8   Pt,Pt0,Pi,zi,time,rsave,Psave
      real*8 deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      real*8  delta,error,slope
      real*8 zip,zim
      real*8 Pip,Pim
      real*8 Ptp,Ptm
      real*8 Mcp,Mcm
      real*8 Mbp,Mbm
      real*8  zierr,Pterr,zisave,Ptsave
      real*8 Mbnow,Mcnow
*
      real*8 Mc,Mb
*
      integer icount,icount2,icount3
      real*8 Pcrude
      real*8 MassCrude,MassBrine
*
      external MassCrude
      external MassBrine
*
      common/startups/  Pw,Rft,zt,zb,zi0,deltat,tquit,tdump,
     &                  DtMb,bcleak,ecleak,LPtConst,Lterm,
     &                  Bcremove,spgr,Klitho,E,mu,A_yr,
     6                  Elast,nu,mT,bT,nrpts,bigX,bigY,
     6                  Mbblb,Mbblc,CcP_psia,CcT_K,Ccrg_cm3,Crho,CP,
     &                  CT,CT2,CT3,alphac,alphab
      common/integ/     xinteg,winteg
      common/variables/ Pt,Pt0,Pi,zi,time,rsave(25000),Psave(25000),
     &                  deltazr,Mbremove,Mbremovet,Mcremove,Mcremovet
*
      zierr = 1.0
      Pterr = 1.0
*
      zisave = zi
      Ptsave = Pt
```

```
*
      icount = 0
*
100   continue
*
* set a delta on zi, put it at 0.5 ft
*
         delta = 0.5/Rft
         error = 1.0
*
* we need Pi for the brine iteration (varying Pt), so
*  use the Pcrude equation at z = zi to produce it
*
         icount2 = 0
200      continue
            zip = zi + delta/2.0
            Pip = Pcrude(zip)
            Pi = Pip
            call  qgaus(MassBrine,zip,zb,Mb)
            Mbp  -Mb

            zim = zi - delta/2.0
            Pim = Pcrude(zim)
            Pi = Pim
            call  qgaus(MassBrine,zim,zb,Mb)
            Mbm - Mb

            slope = (Mbp - Mbm)/(zip - zim)
            zi = (Mbnow - Mbp + slope*zip)/slope
            error = abs(Mbnow - (Mbp + Mbm)/2.0)
            icount2 = icount2 + 1
            if (icount2 .gt. 100) then
               write(6,*)'Subroutine Iterate'
               write(6,*)'100 iterations on Mb (zi) have occured'
               write(6,*)'set error = 0.0 and continue'
               error = 0.0
            endif
         if (error .gt. l.Oe-6) goto 200
*
         zierr = abs(zisave - zi)
         zisave = zi
*
*  set delta on Pt, 1 psia
*
         delta = 0.5*CcP_psia
         error = 1.0
*
         icount3 = 0
*
300      continue
            Ptp = Pt + delta/2.0
            Ptm = Pt - delta/2.0
*
            Pt = Ptp
```

```
            call qgaus(MassCrude,zt,zi,Mc)
            Mcp = Mc

            Pt = Ptm
            call  qgaus(MassCrude,zt,zi,Mc)
            Mcm = Mc
*
            slope = (Mcp - Mcm)/(Ptp - Ptm)
            Pt = (Mcnow - Mcp + slope*Ptp)/slope
*
            error = abs(Mcnow - (Mcp + Mcm)/2.0)
            icount3 = icount3 + 1
            if (icount3 .gt. 100) then
               write(6,*)'Subroutine Iterate'
               write(6,*)'100 iterations on Mc (Pt) have occured'
               write(6,*)'set error = 0.0 and continue'
               error = 0.0
            endif
         if (error .gt. 1.0e-6) goto 300
*
         Pterr = abs(Ptsave - Pt)
         Ptsave = Pt
*
         icount = icount + 1
         if (icount .gt. 100) then
            write(6,*)'Subroutine Iterate'
            write(6,*)'100 iterations on zi & Pt have occured'
            write(6,*)'zierr and Pterr ',zierr,Pterr
            write(6,*)'return to main program'
            return
         endif
      if (zierr .gt. 0.01/Rft .or. Pterr .gt. 0.01*CcP_psia) got0 100
*
      return
      end
* * *
      subroutine  qgaus(func,a,b,ss)
*
* This is the first of two subroutines
*  used to perform the numerical integrations
*  required.
*
*     implicit none
      real*8 xinteg,winteg
      dimension xinteg(5),winteg(5)
*
      real*8 xm,xr,dx
      real*8 func
      real*8  a,b,ss
      integer j
*
      common/integ/     xinteg,winteg
*
      xm = 0.5*(b + a)
```

```
      xr = 0.5*(b - a)
      ss = 0.0
*
      do 100 j = 1,5
         dx = xr*xinteg(j)
         ss = ss + winteg(j)*(func(xm+dx) + func(xm - dx))
100   continue
*
      ss = xr*ss
*
      return
      end
* * *
      subroutine qgaus2(func,a,b,ss)
*
* This is the second of two subroutines
*  used to perform the numerical integrations
*  required.
*
*     implicit none
      real*8 xinteg,winteg
      dimension xinteg(5),winteg(5)
*
      real*8  xm,xr,dx
      real*8 func
      real*8  a,b,ss
      integer  j
*
      common/integ/     xinteg,winteg
*
      xm = 0.5*(b + a)
      xr = 0.5*(b - a)
      ss = 0.0
*
      do 100 j = 1,5
         dx = xr*xinteg(j)
         ss = ss + winteg(j)*(func(xm+dx) + func(xm - dx))
100   continue
*
      ss = xr*ss
*
      return
      end
```

Appendix B

Variable Definitions and Nondimensionalization

Although the definitions and nondimensionalization of the simplified creep model's variables have been documented in the source code (Appendix **A),** these topics are addressed more completely in this appendix. In particular, all **"sprcreep.l"** input variables except those involving time, are nondimensionalized immediately after the program reads them. The program **procedes** in nondimensional form, redimensionalizing the output when it is written to the output files. The program's variables are declared at the top of each routine in the order in which they appear in COMMON blocks. As the COMMON blocks are organized by variable usage, it is convenient to discuss the program's variables as they appear in the COMMON blocks. Any variables which are local to a given routine (i.e. not included in any COMMON block) are declared in that routine immediately following the declarations of the COMMON block variables. In the text and tables below, several references are made to equations detailed in a previous report which documents the mathematics underlying the simplified creep model.[1] These equations are referred to with a I preceding the equation number from the first report (e.g.. (I-13)).

COMMON Block "startups"

The subroutine "start" reads and nondimensionalizes the dimensional parameters from the **"sprcreep.l"** input file. This nondimensionalization is performed using several nondimensionalization variables. Once the **"sprcreep.l"** input variables have been nondimensionalized, these variables as well as the nondimensionalization variables are made available throughout the program via the COMMON block "startups." The definitions and dimensions of these variables have been tabululated in Table B-l.

Table B-l

Variable	Variable Description	Variable Dimension
Pw	initial (t = 0) **wellhead** pressure	psia
Rft	cavern radius	ft
zt	distance from surface to the top of the cavern	ft
zb	distance from surface to the bottom of the cavern	ft

z_i0	initial (t = 0) distance from surface to the oil/brine interface	ft
deltat	time step	yr
tquit	final time	yr
tdump	time interval at which r(z) is recorded in output files "sprcreep.3" and "sprcreep.8"	yr
DtMb	brine removal time interval	yr
bcleak	time at which to begin leaking oil at a rate of Bcremove bbl/time step	yr
ecleak	time at which to end oil leak	yr
LPtConst	logical variable: if true (T), program calculates brine removed every DtMb years necessary to bring P_w back to its original value	logical
Lterm	logical variable: if true (T), program prompts user (at terminal) for oil and brine removal information, if false (F), program either reads necessary information from an input file (if LPtConst = F) or the information is unnecessary (LPtConst = T)	logical
Bcremove	barrels of oil to be leaked per time step	barrels
Klitho	creep equation constant, equation (1)	psia/ft
E	creep equation constant, equation (1)	K
mu	creep equation constant, equation (1)	psia
spgr	oil specific gravity	nondimensional
A_yr	creep equation constant, equation (1)	1/yr

Elast	elastic creep parameter, equation (2)	psia
nu	elastic creep parameter, equation (2)	nondimensional
mT	salt temperature profile constant, equation (3)	**K/ft**
bT	salt temperature profile constant, equation (3)	K
nrpts	number of points in the array used to store r(z)	nondimensional
bigX	nondimensional constant used in the crude oil equation of state, defined as **"M"** in equation (I-18)	nondimensional
bigY	nondimensional constant used in the crude oil equation of state, defined as "B" in equation (I-19)	nondimensional
Mbblb	the mass of a barrel of brine	nondimensional
Mbblc	the mass of a barrel of crude oil	nondimensional
CcP_psia	crude oil equation of state parameter C_c^P **(T),** evaluated at the temperature at the top of the cavern, equation (I-8)	l/psia
CcT_K	crude oil equation of state parameter C_c^T, equation (I-7)	**1/K**
Ccrg_cm3	crude oil equation of state parameter ρ_C°, equation (I-6)	**g/cm^3**
Crho	brine equation of state constant ρ_B°, equation (I-10)	nondimensional
CP	brine equation of state constant C_B^P, equation (I-14)	nondimensional
CT	brine equation of state constant C_B^T, equation **(I-11)**	nondimensional
CT2	brine equation of state constant C_B^{T2}, equation (I-12)	nondimensional

CT3	brine equation of state constant C_B^{T3}, equation (I-13)	nondimensional
alphab	$(\rho_B^\circ)(g)(C_B^P)$(Rft) where g is the gravitational constant, ρ_B° and C_B^P brine equation of state constants, equation (I-10), and Rft the initial radius at the top of the cavern	nondimensional
alphac	$(\rho_C^\circ)(g)(C_C^P(T(z_T))$(Rft) where g is the gravitational constant, ρ_B° a crude oil equation of state constant, $C_C^P(T(z_T))$ a crude oil equation of state constant evalutated at the temperature at the top of the cavern, and Rft the initial radius at the top of the cavern	nondimensional

COMMON Block "integ"

The model requires several numerical integrations to be performed. This is accomplished in two subroutines, "qgaus" and "qgaus2." These routines require two arrays of constants, "xinteg" and "winteg." These arrays are initialized in subroutine "start" and maintained in the COMMON block "integ."

COMMON Block "variables"

This COMMON block contains the program variables which are used in multiple routines but are not "sprcreep.1" input variables. These variables are defined in Table B-2:

Table B-2

Variable	Variable Description	Variable Dimension
Pt	pressure at the top of the cavern, zt	nondimensional
Pt0	initial (t = 0) pressure at the top of the cavern	nondimensional
Pi	pressure at the interface	nondimensional
zi	interface depth	nondimensional

time	time variable	**yrs**
rsave	array which contains the r(z) profile at the previous time **step,** used to determine Ar, equation (1)	nondimensional
Psave	array which contains the P(z) profile at the previous time **step,** used to determine AP, equation (2)	nondimensional
deltazr	spacing between points in the **"rsave"** and "Psave" arrays	nondimensional
Mbremove	mass of the brine to be removed from the cavern at the current time step	nondimensional
Mbremovet	total mass of brine removed from the cavern since time t - o	nondimensional
Mcremove	mass of the crude oil to be removed from the cavern at the current time step	nondimensional
Mcremovet	total mass of crude oil removed from the cavern since time t = 0	nondimensional

COMMON Block "brine"

The variables in this COMMON block are constants in the brine equation of state. Each is initialized and nondimensionalized in subroutine "start." Variables "phi," **"bigI," "bigII,"** and **"bigIII"** are defined by equations (I-27), (I-29), (I-30), and (I-31), respectively.

COMMON Block **"outputc"**

The variables "Mc," "Mb," **"Vc,"** and **"Vb,"** represent the mass of the cavern's crude oil, the mass of the cavern's brine, the volume of the cavern occupied by the crude, and the volume of the cavern occupied by the brine, respectively. These variables are nondimensional and used by the "output" subroutine.

Non-COMMON Block Variables

Several other variables which are not included in the COMMON blocks but which deserve mention are listed in Table B-3. Although most routines in

the program employ local variables, the variables listed in Table B-3 occur in the main routine.

Table B-3

Variable	Variable Description	Variable Dimension
DtMc	time interval for removal of crude oil (leak) when program is operated in the interactive mode	yrs
MbtO	initial (t = 0) mass of brine in the cavern	nondimensional
MctO	initial (t = 0) mass of crude oil in the cavern	nondimensional

Nondimensionalization Multipliers

Finally, the dimensions of the program's variables as well as the parameters used to nondimensionalize these variables have been tabulated in Table B-4.

Table B-4

Dimension	Nondimensionalization Multipliers	Dimensions of Multipliers
psia^n	$\left(C_C^P(T(z_T))\right)^n$	$\left(\frac{1}{\text{psia}}\right)^n$
ft^n	$\left(\frac{1}{R}\right)^n$	$\left(\frac{1}{\text{ft}}\right)^n$
$\left(\frac{1}{K}\right)^n$	$\left(\frac{1}{C_C^T}\right)^n$	$\left(\frac{1}{1/K}\right)^n$
g	$\left(\frac{1}{\rho_C^\circ}\right)\left(\frac{1}{R}\right)^3$	$\left(\frac{1}{\text{g/cm}^3}\right)\left(\frac{1}{\text{ft}}\right)^3$
$\frac{g}{cm^3}$	$\left\lvert\frac{1}{\rho_C^\circ}\right\rvert$	$\left\lvert\frac{1}{\text{g/cm}^3}\right\rvert$
bbl	Mbblc $\left\lvert\frac{1}{\rho_C^\circ}\right\rvert\left\lvert\frac{1}{R}\right\rvert^3$	$\left(\frac{g}{\text{bbl}}\right)\left(\frac{1}{\text{g/cm}^3}\right)\left(\frac{1}{\text{ft}}\right)^3$

In this table, n represents the integer necessary to accomplish nondimensionalization, $C_C^P(T(z_T))$, CCT, and ρ_C° are parameters from the crude oil equation of state, R is the initial radius at the top of the cavern, and Mbblc is the mass of a barrel of crude oil.

Distribution:

US DOE SPR PMO (4)
900 Commerce Road East
New Orleans, IA 70123
Attn: TDCS (2)
E. E. Chapple
D. W. Whittington

USDOE SPR PMO (2)
1000 Independence Avenue SW
Washington, DC 20585
Attn: D. Johnson
D. Smith

Boeing Petroleum Services, Inc. (3)
850 S. Clearview Parkway
New Orleans, LA 70123
K. E. Mills
K. D. Wynn
T. Eyermann

6000 V. L. Dugan
6200 B. W. Marshall
6250 P. J. Hommert
6253 D. S. Preece
6257 J. K. Linn (10)
6257 G. S. Heffelfinger (10)
6257 J. T. Neal
6257 J. L. Todd
6257 B. L. Ehgartner
6257 P. S. Kuhlman
1514 H. S. Morgan
3141 S. Landenburger (5)
3151 G. C. Claycomb
For DOE/TIC (Specified External Distribution Only)
8523 R. C. Christman (Library)